M-SEC-905

Offensive Security and exploitation Pentest Report

Marian Bret

Link to this pdf document as a google doc:
https://docs.google.com/document/d/1NjqQ03sR_aSHPFMjDfkltyNo6yXaJ01LoOF0BeNfH9w/edit#

Link to the Github: https://github.com/marianbret/offensive_security_tek5

Table of Contents

Methodology

Work methodology :

We tried to keep the same methodology for every single information gathering or exploitations we did on the scope 10.10.10.0/24.

The recommended methodology about pentesting is this pattern : **Footprinting > Network scanning > Enumeration > Exploitation**.

Footprinting

First part of the work was to scan the network scope to find the first useful information : the hosts that were up and ports that were open on those hosts.

As we spotted some DNS open ports, we decided to scan them to gather as much information as possible (logs available in 'log-10.10.10.10-53' and 'log-10.10.10.11-53'). Those scans allowed us to get some useful information about hosts and PTR (which are "reverse DNS" that give the domain name associated with an IP). Those information was crucial for deeper scans and possible exploitations.

Network scanning

After getting the whole list of connected hosts and open ports that belong to those hosts, we continued to follow the methodology of pentesting which led to some deeper network scans on a specific host and port (the list is available on 'scan-logs'). We usually used Kali Linux's built-in scanners softwares such as **Nmap** (https://nmap.org/) or **Dirb** for http ports as an example. Those scanners have multiple parameters and flags that allow them to do a specific and precise scan to gather information such as services that are running currently on the network with their specific versions, actual configurations, responses of the server…

During those scans, we used some built-in scripts in scanning softwares such as **nmap-script-engine** (https://nmap.org/man/fr/man-nse.html) or **Metasploit** scanners that can lead to leak of some impacting informations on the victim host (either with the -sC Nmap flag that launch the default script for a port or either precise scripts, using the —script= flag on Nmap).

This scanning work gave us a good overview of the network scope and possible entries vectors to look for possible vulnerabilities deeper.

Enumeration

Keeping on following the pentest methodology, we tried to enumerate as much information as possible on a host/port we found open earlier. To achieve this, we used some softwares and databases such as **searchsploit**, **exploit-db**, **CVE-details** and built-in enumeration scripts (such as **SMB enumeration** or **enum4linux** for example).

Exploitation

With all those scanning and enumerating information, we can try to find the good attack vector on a specific host. At this point, we got two options, either the host/port is vulnerable and we managed to exploit it to find some very impacting information or established a root connection (or vertical/horizontal privilege escalation) or either we found some vulnerabilities and security flaws that haven't been exploited but can led to future exploits. As long as we didn't fulfill in breaking the protections, we continued to follow the methodology in a way to do deeper scans or enumerate more useful information about the system. In fact, we understood that a machine or service we tried to break that led to nothing doesn't mean that the service isn't vulnerable or dangerous for the system (as an example, a service can leak no information but can be vulnerable for Dos attack or else).

The structure we used to store the information we gathered and our exploits is available on the project Github (https://github.com/marianbret/offensive_security_tek5) with this actual PDF report and some logs that follow the same pattern : log-host-port.txt (for example: log-10.10.10.10-22.txt). Those logs contain the scanning information, the enumerations and exploitations.

Scanning

Scanning the project scope:

The project scope is : 10.10.10.0/24

The first machine of the scope is : 10.10.10.1

The last machine of the scope is : 10.10.10.254

The scope scanning logs are available on Github in the ‘scans-logs.txt’ file.

Scanning SSH ports:

The SSH ports scanning are available on Github in the ‘scans-ssh-logs.txt’ file.

During those scans, we spotted that some host-keys are duplicated ones on different machines, that could be a huge risk as an attacker which would have compromised one of those machines could easily use shared ssh keys found in the victim machine to attack other machines with the same host-keys in MITM attacks for example.

Scanning DNS ports :

The DNS ports scanning are available on Github in the ‘scans-logs.txt’ file in the DNS scans section.

Vulnerabilities

10.10.10.22:139:445 :

During the scanning process, we found open ports 139 and 445 on the machine 10.10.10.22. Those ports are SMB ports which is an ‘interprocess-communication protocol’ on Windows. Here it uses Samba which is a software that implements SMB protocol on Unix systems.

We scanned those ports deeper and found that **SMB v1 , v2 and v3 were available** on the server which can lead to some impacting vulnerabilities that can be related to one of each versions.
Nmap scans are available on ‘log-10.10.10.22-139-445’ and gave us some useful information. This version is vulnerable for **regsvc-dos exploit** which leads to a possible Ddos attack that will crash the system.

Nmap smb-enum-shares script, enum4linux and smbmap softwares gave us a lot of information such as the **Anonymous access rights** on the Public directory, the **users** on the system (myles and guest), the **OS information** and **folders tree**.

We succeeded in connecting to the Public shared folder (all exploitations are available in the log linked above) and found multiple files. One of those was a **password protected zip file**, we used the “get” command to fetch all the files from the smb shared drive to our local machine.

With the help of the zip2John software, we created a hash file from the zip archive and let JohnTheRipper crack the password with the built-in rockyou wordlist file (fcrackzip software works perfectly too).

The zip file was cracked, while trying to run the pmanager file, we found that we need some information to fulfill file’s parameters that were an **username** (myles, we had the information by enum4linux) and a **UserId** (that was in plain text in myles’s card picture) we had those data in the enumeration part of the work on this port. The pmanager script returned a **password** for myles user that we used to connect to myles’s smb profile successfully. We found some nice information on this smbclient, a todo list which talks about **vulnerabilities** on thermostat machines and a **ssh private key** which is encrypted with Ccrypt software (we didn’t try any bruteforce on this encryption).

Advise

We strongly advise the administrator to protect the Public directory on the Smb server or to avoid impacting information such as the pmanager.zip or myles png card on a fully access directory.

10.10.10.53:21 :

During the scanning process, we found open port 21 on the machine 10.10.10.53. The port 21 is a 'FTP' port which is a 'file transfer protocol'.

We scanned this port deeper and found out that this actual ftp service allows **Anonymous login** (nmap built-in default script for ftp port) and the version of the service is **vsftpd 2.3.4**.
We managed to connect to the host as an anonymous user without any passwords and had the READ right on it. The server as anonymous was empty, nevertheless, we advise the administrator to connect to Cpanel and uncheck "Anonymous access" and "Anonymous upload" boxes, in a way to avoid this kind of connection.

In the information gathering process, we found with the help of searchsploit tool (which is an exploit database researcher tool) that this precise version of vsftpd is vulnerable with a **backdoor (CVE-73573)** attack which led to a **root shell** on the system (built-in msfconsole or in exploit-db as a python script). We ran the exploit and managed to get the root access (all scans and exploitations are available on 'log-10.10.10.53-21'), we dumped the **/etc/passwd** and **/etc/shadow** files to crack the passwords with **JohnTheRipper** tool and managed to get a credential which is **fern11:naruto1**. We got some **ssh configuration** and **ssh keys** too (public and private). It was accessible through the backdoor exploit root shell, on ftp connecting as fern11 or while connecting as fern11 user on ssh 22 port on the same machine.

Advise

To avoid this kind of exploit, we advise the administrator to update the ftp service from vsftpd 2.3.4 to the newest versions.

10.10.10.222:80 :

During the scanning process, we found open port 80 on the machine 10.10.10.222. The port 80 is an HTTP port. This http service is **Apache httpd 2.4.38 with Wordpress 5.2.4** as an http generator.

We scanned this port deeper and found some useful information (logs and exploits are available in 'log-10.10.10.222-80') such as available **robots-txt** file, plugins used by the WordPress (which is Akismet, we got it through http-wordpress-enum on Nmap script Engine) and users (which is **Fraser**, we got it through http-wordpress-users Nmap script).

For further enumerations, we used tools such as Dirb and Wpscan which gave us information such as the listing directories of the WordPress, the availability of **XMLRPC.php** file, the availability of wp-cron, WordPress themes… With all that information about the host and the system, we searched for vulnerabilities on those versions and we found some with searchsploit such as the plugin **Akismet** that suffer from **multiple cross-site-scripting vulns**.

While enumerating, we spotted a very dangerous vector attack on the XMLRPC.php which is an RPC protocol to share methods and commands through the network. A GET http call on xmlrpc.php is enough to get the information that the server is allowed to receive **POST** requests and try the exploitation. We began with listing the different methods accepted by the xmlrpc protocol on the server using the command 'system.listMethods' in the post data which has returned all the available methods.

Inside this list, we found a few methods that could lead to huge security flaws such as '**wp.getUsersBlogs**', '**wp.getCategories**', '**metaWeblog.getUsersBlogs**' that are vulnerable from login **Bruteforcing** since those methods don't have any connection attempts limit. The other method that can harm seriously the system is '**pingback.ping**', the attacker could use this one to achieve **XSPA** (cross-site port attacks) exploits or **Ddos** attacks sending huge pingbacks with an http server.

Advise

We advise the system's administrator to disable the xmlrpc protocol on this site since a new REST API has been released for WordPress and contains much less vulnerabilities and to update the Akismet's plugin to versions above 3.1.5 to avoid those multiple XSS flaws.

10.10.10.10:53 :

During the scanning process, we found open port 53 on the machine 10.10.10.10. The port 53 is a domain port on TCP/UDP protocol and runs a DNS service.

We scanned this port deeper (logs and exploits are available in 'log-10.10.10.10-53') and found that the DNS service was running under dnsmasq-2.75.
We used the nslookup tool to enumerate some information about this DNS service and we found that the linked domain name was '**dns1.powerzio.lan**'.
With this information, we tried further enumeration with DIG (which is a dns client for unix) and the response returned the **'RA'** flag that stands for '**Recursion available**' that enables recursive DNS but can be considered as a flaw since attackers can use this recursivity to launch **Ddos or cache poisoning attacks.**
The most useful information we got on this service has been found with the dnsrecon tool (DNS tool that allows a lot of dns requests such as zone transfer, dns enumeration…) with PTR reverse lookups that are the opposite of 'A' dns records, PTR records return the domain name of an IP or an IP range, we've been able to find **17 PTR records** on the range with useful information about other network services.

10.10.10.11:53 :

During the scanning process, we found open port 53 on the machine 10.10.10.11. The port 53 is a domain port on TCP/UDP protocol and runs a DNS service.

We scanned this port deeper (logs and exploits are available in 'log-10.10.10.11-53') and found that the DNS service was running under dnsmasq-2.75.
We used the nslookup tool to enumerate some information about this DNS service and we found that the linked domain name was '**dns2.powerzio.lan**'.
With this information, we tried further enumeration with DIG and the response returned the **'RA'** flag that stands for '**Recursion available**' that enables recursive DNS but can be considered as a flaw since attackers can use this recursivity to launch **Ddos or cache poisoning attacks.**

The most useful information we got on this service has been found with the dnsrecon tool with PTR reverse lookups. We've been able to find **12 PTR records** on the range with useful information about other network services.

10.10.10.48:80 :

During the scanning process, we found open port 80 on the machine 10.10.10.48. The port 80 is an HTTP port. This http service is nodeJS with Express framework.

We scanned this port deeper (logs and exploits are available in 'log-10.10.10.48-80') and found that http-cors allows **HEAD GET POST PUT DELETE PATCH** http methods on the server.
While looking at the http page source code, we spotted that a button was calling the endpoint /api/config with a **POST** method and the Dirb scanner returned two similar GET endpoints (/api/temp or /api/TEMP).

In a way to enumerate as much information as possible, we used a known web analysis tool called **ZAP** from OWASP society. ZAP is an open source web security scanner which is pretty complete. This scan returned **7 alerts** (complete report is available on the github in 'report-10.10.10.48.html', please open it as an HTML file) of which one is **high risk**, three are medium risks and three are low risks.

The highest risk vulnerability found with the scanner is a **remote OS command injection** through /api/config with POST http requests (<script type="text/javascript">window.location="/";</script>). This attack is possible as the **server accepts untrusted input and data through this endpoint.**
Some misconfigurations could lead to possible exploits such as Cross-domain or X-Frame options header that aren't set to any values or that are misconfigured.

Advise

In a way to reduce risks, even more for remote os command injection, we advise the administrator to prefer using some library calls instead of external calls that could be used differently and then could be a huge flaw on the server. About the Cross-domain misconfiguration, we advise a complete check of data and its sensitivity to be sure any important data couldn't be accessible with unauthenticated endpoints/calls.

10.10.10.55:80 :

During the scanning process, we found open port 80 on the machine 10.10.10.55. The port 80 is an HTTP port. This http service is nodeJS with Express framework.

We scanned this port deeper (logs and exploits are available in 'log-10.10.10.55-80') and found that http-cors allows **HEAD GET POST PUT DELETE PATCH** http methods on the server.
While looking at the http page source code, we spotted that a button was calling the endpoint /api/config with a **POST** method and the Dirb scanner returned two similar GET endpoints (/api/temp or /api/TEMP).

In a way to enumerate as much information as possible, we used the ZAP tool from OWASP society. This scan returned **7 alerts** (complete report is available on the github in 'report-10.10.10.55.html', please open it as an HTML file) of which one is **high risk**, three are medium risks and three are low risks.

The highest risk vulnerability found with the scanner is a **remote OS command injection** through /api/config with POST http requests (<script type="text/javascript">window.location="/";</script>). This attack is possible as **the server accepts untrusted input and data through this endpoint**.
Some misconfigurations could lead to possible exploits such as Cross-domain or X-Frame options header that aren't set to any values or that are misconfigured.

Advise

In a way to reduce risks, even more for remote os command injection, we advise the administrator to prefer using some library calls instead of external calls that could be used differently and then could be a huge flaw on the server. About the Cross-domain misconfiguration, we advise a complete check of data and its sensitivity to be sure any important data couldn't be accessible with unauthenticated endpoints/calls.

10.10.10.53:22 :

During the scanning process, we found open port 22 on the machine 10.10.10.53. The port 22 is an SSH port which stands for secure shell and it's a secure file transfer protocol with strong password and encryption keys management. Keys (public and private) from both sides are required to ensure the security of this protocol.

We scanned this port deeper (logs and exploits are available in 'log-10.10.10.53-22') and found that SSH was running under OpenSSH7.2. We've been able to directly **connect** on this ssh port as we managed to find the **fern11 user password** on 10.10.10.53:21 exploitation.
As for the 21 ftp port on the same machine, we've **dumped both the private and public ssh key.**

Advise

We advise the administrator to use key pairs connection instead of simple password connection and as the same advice for the ftp port on this IP, we strongly advise the administrator to update ftp service version to avoid easy backdoor exploits.